Python JSON

Seguinte "

JSON é uma sintaxe para armazenamento e troca de dados.

JSON é um texto, escrito com JavaScript object notation.

JSON em Python

Python tem um built-in pacote chamado `json`, que pode ser usado para trabalhar com dados em formato JSON.

Exemplo

de Obter o seu próprio Python Server

Importar o json módulo:

```
import json
```

Analisar JSON - conversão de JSON para o Python

Se você tem uma seqüência de caracteres JSON, você pode analisá-lo usando o `json.loads()` o método.

Exemplo

Converter de JSON em Python:

```
import json

# some JSON:
x =  '{ "name":"John", "age":30, "city":"New York"}'

# parse x:
y = json.loads(x)

# the result is a Python dictionary:
print(y["age"])
```

Tente você Mesmo "

Converter a partir de Python para JSON

Se você tem um objeto Python, você pode convertê-lo em uma seqüência de caracteres JSON por usando o `json.dumps()` o método.

Exemplo

Converter a partir de Python para JSON:

```
import json

# a Python object (dict):
x = {
```

```
# convert into JSON:
y = json.dumps(x)

# the result is a JSON string:
print(y)
```

Tente você Mesmo "

Você pode converter de Python, objetos dos seguintes tipos, no JSON cadeias de caracteres:

- dict
- lista
- tuple
- cadeia
- int
- float
- Verdadeiro
- Falso
- Nenhum

Exemplo

Converter Python objetos em JSON cadeias, e imprimir os valores:

```
import json

print(json.dumps({"name": "John", "age": 30}))
print(json.dumps(["apple", "bananas"]))
print(json.dumps(("apple", "bananas")))
print(json.dumps("hello"))
print(json.dumps(42))
```

Tente você Mesmo "

Quando você converter de Python para JSON, Python objetos são convertidos para o formato JSON (JavaScript) equivalente:

Python	JSON
dict	Object
list	Array
tuple	Array
str	String
int	Number
float	Number
True	true
False	false
None	null

Exemplo

Converter um objeto Python contendo todos os meios legais tipos de dados:

```
import json

x = {
  "name": "John",
  "age": 30,
```

Formatar o Resultado

O exemplo acima imprime uma seqüência de caracteres JSON, mas não é muito fácil de ler, sem recuos e quebras de linha.

O `json.dumps()` o método tem parâmetros para torná-lo mais fácil de ler o resultado:

Exemplo

Use o `indent` parâmetro para definir os números de recuos:

```
json.dumps(x, indent=4)
```

Tente você Mesmo "

Você também pode definir separadores, o valor padrão é (", ", ": "), que significa que o uso de uma vírgula e um espaço para separar cada objeto, e uma vírgula e um espaço para separar as teclas de valores:

Use o `separators` parâmetro para alterar o separador padrão:

```
json.dumps(x, indent=4, separators=(". ", " = "))
```

Tente você Mesmo "

Ordem do Resultado

O `json.dumps()` o método tem parâmetros para ordem, as chaves do resultado:

Exemplo

Use o `sort_keys` parâmetro para especificar se o resultado deve ser ordenado ou não:

```
json.dumps(x, indent=4, sort_keys=True)
```

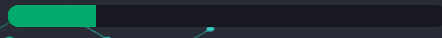
Tente você Mesmo "

Quando você analisar o código com o `json.loads()` método, o resultado é devolvido como um específico de Python tipo de dados, qual?

- lista
- conjunto

Enviar Resposta "

"Anterior Seguinte "

Track your progress - it's free! Sign Up Log in

SELETOR DE CORES